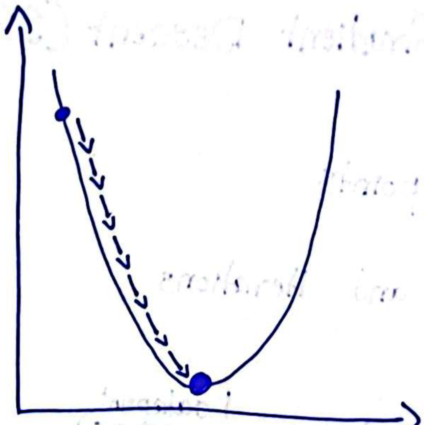
Optimizers

→ Gradient Descent

$$W_{new} = W_{old} - \alpha \frac{\partial L}{\partial W_{old}}$$



MSE

$$\text{Loss function} = (y - \hat{y})^2$$

$$\text{Cost function} = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

Epochs, Iterations

Dataset = 1000 datapoints

Epoch:1 { forward propagation → 1000 datapoint ← backward propagation }

1 Epoch = 1 Iteration

Epoch:2 { → ← }

⋮

Epoch: n { → ← }

> Advantages

① Convergence will happen

> Disadvantage

① Huge resource required such as RAM and GPU

→ Stochastic Gradient Descent (SGD)

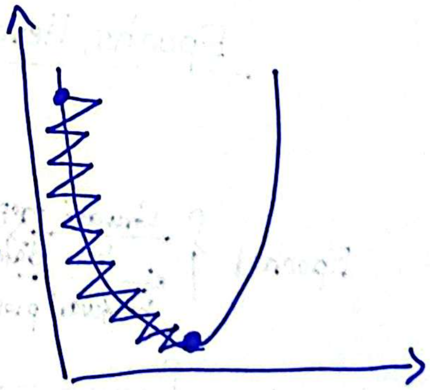
1000 datapoints

Epochs and Iterations.

Epoch 1
- Iteration 1 $\xrightarrow{\text{1 datapoint}}$ ←
- Iteration 2 $\xrightarrow{\text{2 data point}}$ ←
- ⋮
- Iteration 1000 $\xrightarrow{\text{1000th datapoint}}$ ←

> Advantages

① Resource issue solved



> Disadvantages

① Time complexity

② Convergence will take more time

③ Noise gets introduced

→ Mini Batch SGD

Epoch, Iteration, Batch size

Datapoints = 100000

Batch-size = 1000

$$\text{No. of iterations} = \frac{\text{Datapoints}}{\text{batch size}} = \frac{100000}{1000} = 100$$

Epoch 1 : Iteration : 1

1000 datapoints

$$\text{cost function} = \sum_{i=1}^{1000} (y_i - \hat{y}_i)^2$$

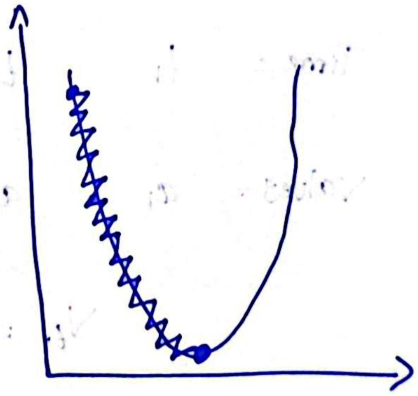
Epoch 1 : Iteration : 2

1000 datapoints

⋮

Epoch 1 : Iteration : 100

1000 datapoints



> Advantages

1. Converges faster than SGD
2. Noise will be less than SGD
3. Efficient resource usage

> Disadvantage

1. Noise still exists

→ SGD with Momentum

$$W_{new} = W_{old} - \alpha \cdot \frac{\partial L}{\partial W_{old}}$$

$$b_{new} = b_{old} - \alpha \frac{\partial L}{\partial b_{old}}$$

$$W_t = W_{t-1} - \alpha \frac{\partial L}{\partial W_{t-1}}$$

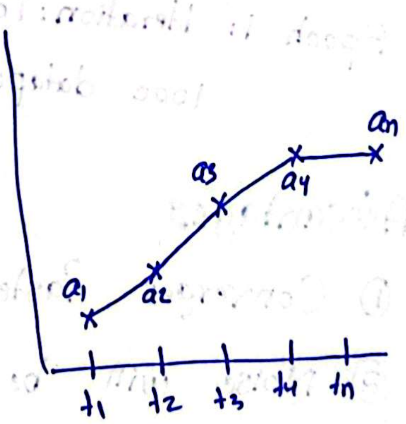
Exponential Weight Average

Use to perform smoothening

Time : $t_1 \quad t_2 \quad t_3 \quad t_4$ ------- t_n

Values = $a_1 \quad a_2 \quad a_3 \quad a_4$ ------ a_n

$$V_{t_1} = a_1$$



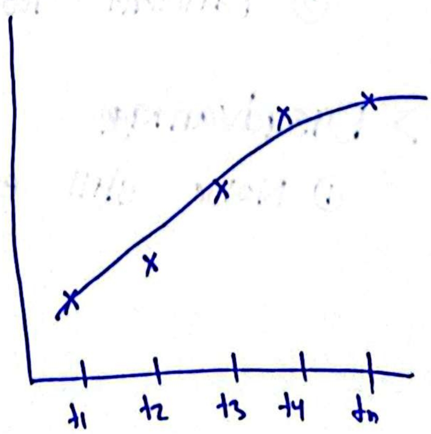
$$V_{t_2} = \beta V_{t_1} + (1-\beta) a_2$$

$$\beta = 0.95$$

$$V_{t_2} = 0.95 a_1 + (1-0.95) a_2$$

$$= 0.95 a_1 + 0.05 a_2$$

> If β has high value, it will control the previous value more.



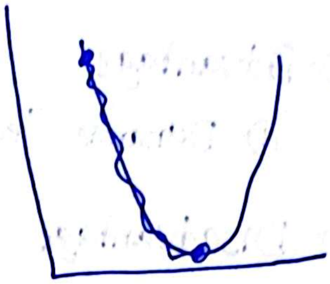
$$V_{t_3} = \beta V_{t_2} + (1-\beta) a_3$$

$$= 0.95 [0.95 a_1 + 0.05 a_2] + 0.05 a_3$$

> Advantage

① Reduce the noise

② Quick convergence



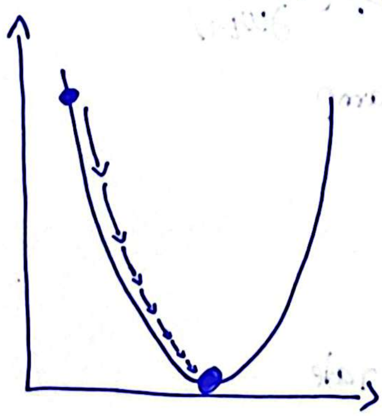
→ Adagrad : Adaptive Gradient Descent

~~W~~ $W_{new} = W_{old} - \alpha \frac{\partial L}{\partial W_{old}}$

α = fixed

⇓

α = dynamic



As the convergences happen, the learning rate will change

$$W_{\cancel{new}\,t} = W_{\cancel{old}\,t-1} - \alpha' \frac{\partial L}{\partial W_{\cancel{old}\,t-1}}$$

$$\alpha' = \frac{\alpha}{\sqrt{\alpha_{t} + \epsilon}}$$ → epsilon

$$\alpha_t = \sum_{i=1}^{t} \left(\frac{\partial L}{\partial W_t}\right)^2$$

$\therefore \alpha_t$ increases, α' decreases

$t=1$	$t=2$	$t=3$
$\alpha=0.01$	$\alpha=0.005$	$\alpha=0.003$

> Advantage:

① Dynamic learning rate

> Disadvantage

① Possibility of learning rate to become approx. zero

② Convergence may never occur

→ Adadelta and RMSPROP

$$\alpha' = \frac{\alpha}{\sqrt{Sdw + \epsilon}}$$

$$Sdw_t = \beta \, Sdw_{t-1} + (1-\beta)\left(\frac{\partial L}{\partial w_{t-1}}\right)^2$$

for the first time stamp

$$Sdw_t = 0$$

> Advantages

① Dynamic learning rate

② Smoothening Exponential Weighted Average

$$w_t = w_{t-1} - \alpha' \frac{\partial L}{\partial w_{t-1}}$$

→ Adam Optimizer

SGD with Momentum + RMSPROP

$$w_t = w_{t-1} - \alpha' V_{dw} \quad \Rightarrow \text{weight updation}$$

$$b_t = b_{t-1} - \alpha' V_{db} \quad \Rightarrow \text{bias updation}$$

$$\alpha' = \frac{\alpha}{\sqrt{S_{dw} + \epsilon}}$$

$$S_{dw_t} = 0$$

$$S_{dw_t} = \beta S_{dw_{t-1}} + (1-\beta)\left(\frac{\partial L}{\partial w_{t-1}}\right)^2$$

$$\left.\begin{aligned} V_{dw_t} &= \beta V_{dw_{t-1}} + (1-\beta)\frac{\partial L}{\partial w_{t-1}} \\ V_{db_t} &= \beta V_{db_{t-1}} + (1-\beta)\frac{\partial L}{\partial b_{t-1}} \end{aligned}\right\} \text{Momentum}$$

↳ Smoothening